# Sentilyzer – A Mashup Application for the Sentiment Analysis of Facebook Pages

**Hartmut Glücker, Manuel Burghardt, and Christian Wolff**
Media Informatics Group
Institute for Studies in Information and Media, Language and Culture
University of Regensburg
firstname.lastname@ur.de

## Abstract

We present *Sentilyzer*, a web-based tool that can be used to analyze and visualize the sentiment of German user comments on *Facebook* pages. The tool collects comments via the *Facebook API* and uses the *TreeTagger* to perform basic lemmatization. The lemmatized data is then analyzed with regard to sentiment by using the *Berlin Affective Word List – Reloaded* (BAWL-R), a lexicon that contains emotional valence ratings for more than 2,900 German words. The results are visualized in an interactive web interface that shows sentiment analyses for single posts, but also provides a timeline view to display trends in the sentiment ratings.

## 1 Introduction

Social media platforms such as *Facebook* or *Twitter* churn out vast amounts of user generated content. This data can be analyzed with regard to subjective information – i.e. people's emotions, attitudes, opinions, and sentiments – to monitor specific topics or detect trends. Such analyses are typically referred to as *sentiment analysis* or *opinion mining* [Liu, 2012].

This article introduces *Sentilyzer*, a web application for the sentiment analysis and visualization of user comments on Facebook pages. The comments are lemmatized and sentiment scores are clustered according to previously defined keywords. The results of the sentiment analysis are presented to the user in an interactive web interface. The rest of the article is structured as follows: Section 2 gives an overview of the technical realization of *Sentilyzer*; section 3 presents the main features and basic functionality of the tool. Section 4 concludes the insights of a first case study that has been conducted with *Sentilyzer*, and also describes the next steps in the development of the prototype.

## 2 Technical realization of Sentilyzer

*Sentilyzer* is realized by means of a client-server architecture that requires an *Apache* server with *PHP* and a *MySQL* database. Lemmatization and sentiment analysis are realized on the server-side by using *Java*. *Sentilyzer* can be categorized as a *mashup* application, as it integrates a number of freely available, third-party components in a common web interface:

**Data crawler and web interface:** *Facebook Graph API* (application programming interface for crawling Facebook data)[1], *Foundation 5.1* (HTML template framework)[2], *Isotope.js 2.0* (JavaScript library for element sorting)[3], *Laravel 4.1* (PHP framework for web applications)[4], *NVD3.js 1.1*



[1] https://developers.facebook.com/docs/graph-api; all URLs mentioned in this paper were last accessed July 10, 2014

[2] http://foundation.zurb.com/

[3] http://isotope.metafizzy.co/

[4] http://laravel.com/

(JavaScript library for facilitated creation of graphs based on the *D3.js* library)[5]

**Lemmatizer and POS tagger:** *TreeTagger* (POS tagger and lemmatizer for German)[6], *TT4J* (Java wrapper for TreeTagger)[7]

**Sentiment lexicon:** *Berlin Affective Word List – Reloaded* (BAWL-R)[8]

## 3 How Sentilyzer works: Basic functionality in five steps

The basic functionality of *Sentilyzer* can be broken down into five basic steps that are explained in more detail in the following sections.

### 3.1 Preliminaries: Project and database setup (Step 1)

Before *Sentilyzer* can analyze the sentiment of Facebook comments, the user needs to define the basic project details via an *XML* configuration file. First, the name of the *Facebook* page that is to be analyzed needs to be specified. Users can also define a timeframe (start and end date) for posts from this page to be included in the analysis. As *Sentilyzer* allows the user to display aggregated sentiment scores for clusters of comments as well as sentiment trends for such clusters throughout time, it is important to specify the parameters for these clusters in advance. It is possible to define arbitrary *timelines* (=clusters of posts) containing only posts that include or exclude certain keywords:

```
<timeline>
 <name>Michael Wendler</name>
  <includePostsWithKeywords>
   <keyword>Michael</keyword>
   <keyword>Wendler</keyword>
  </includePostsWithKeywords>
  <excludePostsWithKeywords>
   ...
  </excludePostsWithKeywords>
</timeline>
```

[5] http://nvd3.org/

[6] http://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/

[7] https://code.google.com/p/tt4j/

[8] http://www.ewi-psy.fu-berlin.de/einrichtungen/arbeitsbereiche/allgpsy/BAWL-R/index.html

After a new project has been created according to the parameters specified in the XML-configuration file, a corresponding database structure is created automatically by the tool.

### 3.2 Crawling the Facebook page (Step 2)

In the second step, the crawler component collects all posts and comments from the previously specified *Facebook* page via the *Facebook Graph API*. The following information for posts and associated user comments is stored in the relational database:

**Posts:** *message text*, *number of likes*, *number of comments*, *number of shares*, *date of publication*

**User comments:** *author name*, *message text*, *number of likes*, *date of publication*

### 3.3 Clustering of posts (Step 3)

In this step the tool creates *timeline* clusters of posts according to the keywords that have been specified in Step 1. This clustering of posts allows the user to compare aggregated sentiment scores of different *timelines* (e.g. for different celebrities) in the final step.

### 3.4 Lemmatization and calculation of sentiment scores (Step 4)

Step 4 contains two important sub-steps: First, the message texts are lemmatized to make them available for automatic sentiment analysis. *Sentilyzer* utilizes an existing lemmatizer for German language, the *TreeTagger* [Schmid, 1994].

Second, the lemmatized comments are compared with a lexicon that contains sentiment scores for different words. For the German language, there are only few resources that can be used as a sentiment lexicon. We identified the *Multi-layered Reference Corpus for German Sentiment Analysis* (MLSA) [Clematide et al., 2012] and the *Berlin Affective Wordlist – Reloaded* (BAWL-R) [Võ et al., 2006, 2009] as appropriate resources for this project. Eventually, we decided to use the BAWL-R lexicon, as it provides more sentiment annotations for single words (over 2,900 words) than MLSA (about 820 words), with the latter being more focused on multi-level sentiment annotation that includes larger units such as *phrases* and *sentences*.


Figure 1: The example shows the original comment and the lemmatized version as well as the BAWL-R sentiment score for a matching word.

The BAWL-R lexicon provides scores for *emotional valence*[9], "ranging from –3 (*very negative*) through 0 (*neutral*) to +3 (*very positive*)" [Võ et al., 2009, p. 535]. The positive and negative values of words that match the BAWL-R lexicon are summed up to an aggregated sentiment score for each comment (cf. Figure 1).

### 3.5 Visualization of sentiment scores (Step 5)

In the last step, the results are visualized in an interactive web interface. The results are organized according to the *timelines* that were specified in Step 3. All posts of a *timeline* are displayed chronologically and can be sorted with respect to different parameters such as *positive / negative sentiment*, *number of comments*, etc. (cf. Figure 2). Alongside the *message content*, *number of likes*, *number of comments*, *number of shares* and *publication date*, the tool displays the aggregated sentiment score for all comments that are associated with a post. The tool also provides an aggregated sentiment score for all comments associated with a specific timeline as well as a view that shows sentiment trends for comments to different posts in the course of time (cf. Figure 3).

## 4 Conclusions and outlook

*Sentilyzer* serves as a proof of concept for a tool that is able to *crawl* user comments from *Facebook* pages, to *analyze* their sentiment, and to *visualize* the results in a user-friendly and interac-

[9]BAWL-R also contains information about *arousal* and *imageability*. This additional information was not utilized in the current prototype, but could be supplemented in a later version of the tool.

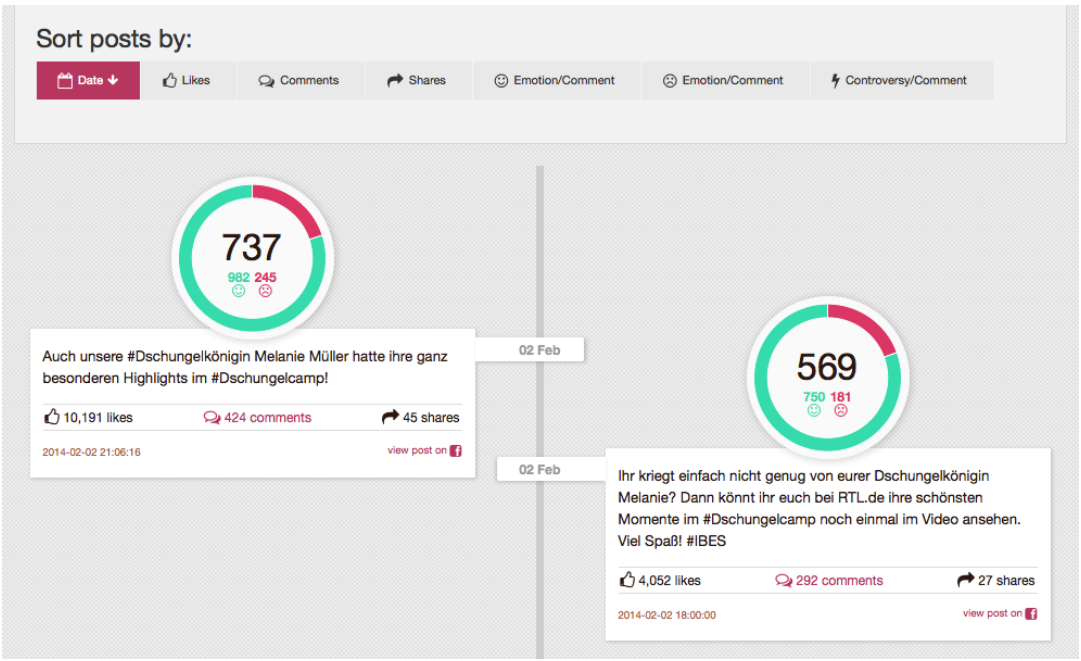


Figure 2: Posts with aggregated sentiment scores for all associated comments. The posts are displayed chronologically be default, but can be sorted by a number of different parameters as well.

tive web interface. As the tool utilizes a number of freely available APIs and tools as well as an existing sentiment lexicon for German, it may be considered a *mashup* application. By using third party components for natural language processing and sentiment analysis of social media data it also becomes obvious that existing resources are not optimized for the specifics of computer-mediated language, e.g. specialized vocabulary and "loose" orthography. We are planning to create a crowd-sourced lexicon with lemmatized forms and sentiment scores for computer-mediated language in an upcoming research seminar on sentiment analysis, thus hopefully improving the current weaknesses of the prototype.

Nevertheless, *Sentilyzer* has already been used successfully to analyze the perception of candidates from the German reality show "Ich bin ein Star - Holt mich hier raus (2014)" on the official Facebook page[10]. The large number of user comments compensated for most of the erroneous lemmatizations and sentiment scores, and could be used successfully to show aggregated sentiment scores and sentiment trends through the course of the TV show.

A live demo of *Sentilyzer* with sentiment visualizations for all candidates is available at `http://dh.wappdesign.net/`. We are currently working on a documented version of the application that will be available via *GitHub* for local installation. In the long-term, we are planning to host *Sentilyzer* as a web service.

[10] `https://www.facebook.com/IchBinEinStar`

## References


Simon Clematide, Stefan Gindl, Manfred Klenner, Stefanos Petrakis, Robert Remus, Josef Ruppenhofer, Ulli Waltinger, and Michael Wiegand. MLSA-A Multi-layered Reference Corpus for German Sentiment Analysis. In *Proceedings of LREC '12*, pages 3551–3556, 2012.

Bing Liu. *Sentiment Analysis and Opinion Mining*. Morgan & Claypool, San Rafael, CA, 2012.

Helmut Schmid. Probabilistic part-of-speech tagging using decision trees. In *International Conference on New Methods in Language Processing*, pages 44–49, Manchester, UK, 1994.

Melissa L H Võ, Arthur M Jacobs, and Markus Conrad. Cross-validating the Berlin Affective Word List. *Behavior research methods*, 38(4): 606–609, 2006. ISSN 1554-351X.

Melissa L-H Võ, Markus Conrad, Lars Kuchinke, Karolina Urton, Markus J Hofmann, and Arthur M Jacobs. The Berlin Affective Word List Reloaded (BAWL-R). *Behavior research methods*, 41(2):534–538, 2009. ISSN 1554-351X.


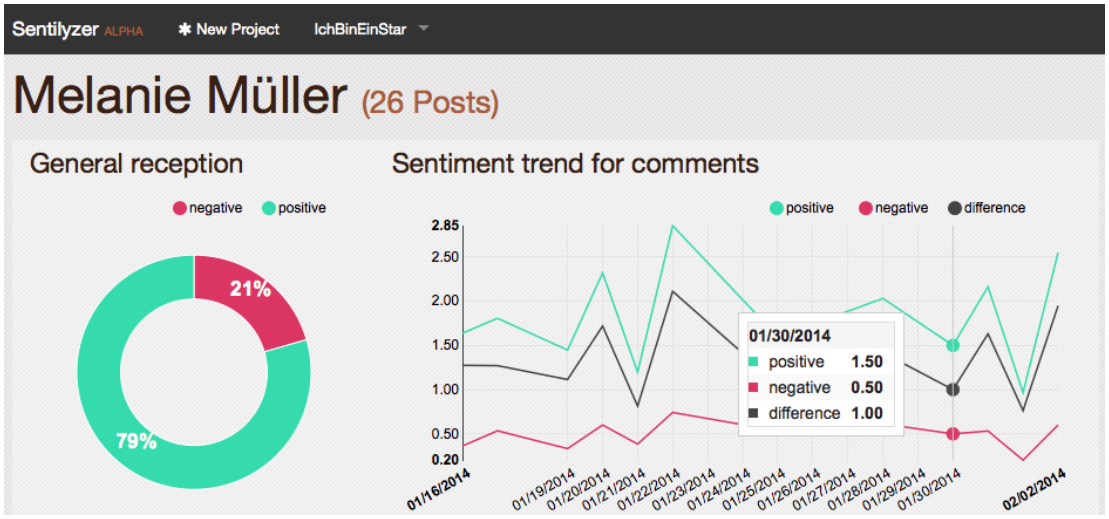


Figure 3: Aggregated sentiment score for all comments associated to a specific timeline and sentiment trend in the course of time.